XML Types for C#

Wolfram Schulte
joint work with
Erik Meijer

Bill Gate wrote recently in a note on "A Software Driven Future" :

"*We need to come up with language extensions that make it easy to program in the XML world.*"

Why is that?

- XML is the lingua franca of the Internet
  - XML is typed content, which helps with interoperability.
- Reliability is crucial in the era of Web services.
  - Correctness, scaleability is a must
- But what is the current state of affairs?
  - A plethora of experimental, often untyped languages, without modul systems, libraries, versioning, ...

Motivation of Further Work

- New X... languages are restricted
  - Own syntax, new/no type system, no module system, limited library...
- Old C... languages are advanced
  - Standard syntax, type system, module system, rich library...

Why is that?

- XML is the lingua franca of the Internet
  - XML is typed content, which helps with interoperability.
- Reliability is crucial in the era of Web services.
  - Correctness, scaleability is a must
- But what is the current state of affairs?
  - A plethora of experimental, often untyped languages, without modul systems, libraries, versioning, ...

Languages and Approaches How to Use and Integrate XML

Language/ Feature	Purpose	Paradigm/ Syntax	Typesystem	Technology	Restrictions
XML Schema	Type Declarations	Data defs/ XML	XML	Validating Parser	Sublanguage
XPath	Projection	Functional/ Directory Paths	None	Interpreter/ Compiler	Sublanguage
XQuery	Query Language	Functional/ Own	XML	Prototype	Domain specific
XSLT	Transformation Language	Functional/ XML	None	Interpreter/ Compiler	Domain specific
XDuce	Explore DTD Typesystem	Functional/ SML	Monomorphic DTDs	Interpreter/ Compiler	Experimental
XMLambda	Explore DTD typesystem	Data defs: Fun/ Haskell + XML	Polymorphic DTDs	Not implemented	Experimental
System.XML	XML support for C#	Imperative/ C#	C#	Library	XML proc. untyped
XSD Compiler	XML support for C#	Data defs: Imp./ C#	C#	Compiler	C#-XML Type mismatch

Motivation of Further Work

- New X... languages are restricted
  - Own syntax, new/no type system, no module system, limited library...
- Old C... languages are advanced
  - Standard syntax, type system, module system, rich library...

The type and paradigm clash!

C languages (C#/Java/VB)

- Imperative, OO
- Simple Lexis
- Global types only
- Two kind of types
  - Value Type
  - Reference Type
- Element and singleton two distinct types
- Subtyping based on
  - Named relationships

X languages (XML/XPath/XQuery)

- Declarative
- Complex lexis
- Global and anonymous types
- Two kind of types
  - Simple types
  - Complex types
- Identification of element and singleton type
- Subtyping based on
  - Inclusion (simple types)
  - Structural Equivalence
  - Named relationships

Solution: XML Types for C#

The proposal is called:

C#-xml

The type and paradigm clash!

C languages (C#/Java/VB)

- Imperative OO
- Simple Lexis
- Global types only
- Two kind of types
  - Value Type
  - Reference Type
- Element and singleton two distinct types
- Subtyping based on
  - Named relationships

X languages (XML/XPath/XQuery)

- Declarative
- Complex lexis
- Global and anonymous types
- Two kind of types
  - Simple types
  - Complex types
- Identification of element and singleton type
- Subtyping based on
  - Inclusion (simple types)
  - Structural Equivalence
  - Named relationships

XML Types and Values are First Class Citizens in C#-xml

- XML types can be denoted
- XML values can be constructed, loaded, passed, transformed, updated, and written in a type safe manner
- This improves reliability and performance

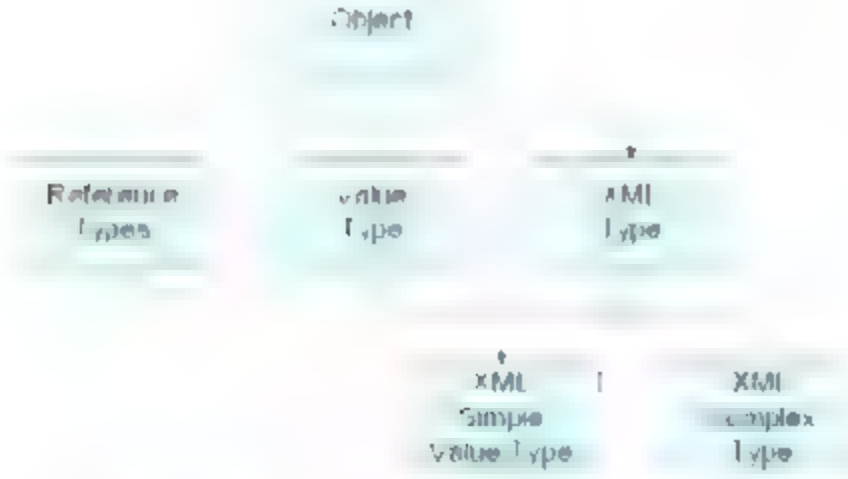
Design: Types

Add XML schemas as C# types

- C# value types stand in for XML *simple types*; implicit conversions replace inclusion
- *Complex types* are new; structural subtyping supported (named subtyping not yet)
- New *sequence type* and iterators are added

- Type system from [illegible]
- [illegible]

The Combined Type System



Example Bib Schema

Type Representation

New typeconstructors

The Bob schema

XML Subtyping (<:)

- Simple Types:
  - Inclusion of value spaces
  - normalizedString <: string
- Complex Types
  - Inclusion of sets of sequences they denote
  - Let t1 t2 be types then
    t1 <: choice{t1 t2} and t1 t2 <: choice{t1 t2}
  - Let t t be types then if t <: t and m <= m and n <= n then
    t[m-n] <: t[m-n]

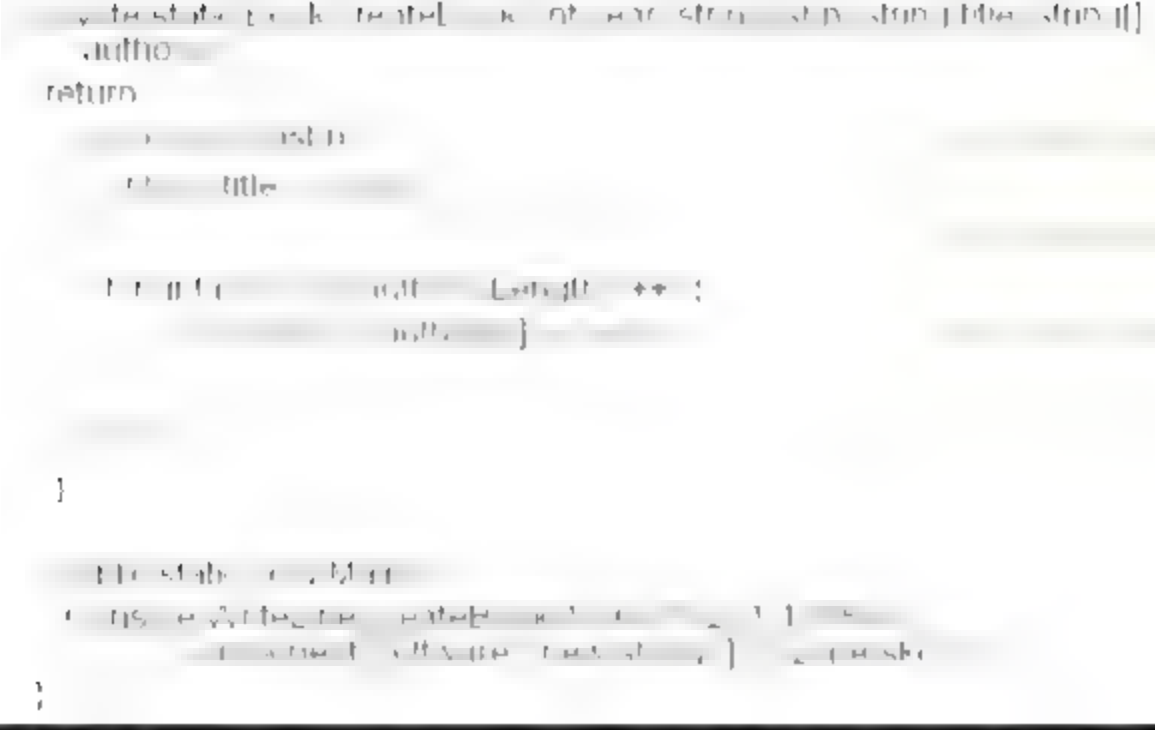
Example Bib Document

```
<?xml version="1.0" encoding="UTF-8" ?>

<!DOCTYPE bib SYSTEM "http://www.oreilly.com/catalog/bib/1.0/bib.dtd">

<bib>
    <book isbn="0-201-63361-2">
        <title>Essential XML</title>
        <author>Box</author>
        <author>Skonnard</author>
        <author>Lam</author>
    </book>

    <book isbn="0-201-63460-0">
        <title>Applied XML</title>
        <author>Hunter</author>
    </book>
</bib>
```

XML Namespaces and Literals

Multiple and Default Namespaces

XML Quotes



Projection

More Projections

Updates

New Statements

```
set x[ ... ] = exp1
```

Select the node (sequence) whose content is updated

here update a book attribute

```
xml [ ... ] [ ... ] = ...
```

C#-xml may introduce runtime check to guarantee type correctness. Same as the array update problem in C#

More Updates

- [illegible]
- [illegible]
- *Add book b to bib database before the author Box*
- [illegible]

Updates: Parameter passing

XML values are math values, therefore

- Only (parts of) variables containing XML values can be updated
- For updates parameter must be passed by reference

```
stat c inc (  ' Book b) {
```

Iteration, Join

Extended **foreach** statement

Select from two databases all books that have overlapping authors but different titles

Book [0-*] res

Input/Output

Extended typed IO

```
using Bib.xsd
using System.XML

public static void Main(string[] args) {
    Bib bibIn = Bib.Load("bib.xml");
    Bib bibOut = Bib.Load("bib.xml");

    foreach (Bib.bibRow bibRow in bibIn.bib.Rows) bibOut.bib.AddbibRow(bibRow);

    bibIn.Close();
    bibOut.Save("bib2.xml");
}
```

Stream-Iterators

Iterators for processing information piece by piece

```
static          Book[0 *] StripPrice .Book[0 *] is1 {
  foreach (Book b in is1
        ' book {b/@isbn} ' {b/title}{b/author} ' /book '
}
```

Summary

X# extends C# by

- Full support of XSD
  - But XML values live in separate value space
- More than XQuery
  - But imperative
- A little bit for Streamprocessing
  - But can hopefully be extended

Thanks for coming!

Most recent version can be found at
http://msrweb/fse/schulte/XMLIntegration.doc

Input/Output

Extended typed IO:

```
using Bib.xsd;
using System.XML;

public static void Main(string [] args) {
   readonly Bib bibIn = Bib.OpenRead(arg[0]);
   writeonly Bib bibOut = Bib.OpenWrite(arg[1]);

   bibOut = <bib> {StripYearAndPrice(get bibIn/Book)} </bib>;

   bibIn.Close();
   bibOut.Close();
}
```

Thanks for coming!

Most recent version can be found at
http://msrweb/fse/schulte/XMLIntegration.doc